Amendments to the Specification:

Please amend the paragraph beginning on line 3 of page 14 of the application to read as follows:

The transport manager will now be described with reference to Fig. 10 which illustrates an overview of transport data management. The transport manager is preferably a software module deployed at the encoding facility 28 or other facility designated as a NOC. As shown in Fig. 10, multiple data sources ~~14~~ 24 (e.g., database content, programs and applications) provide content as input into the transport manager 170. Information regarding the content from these data sources is also provided to the transport manager such as identification of input source ~~14~~ 24 and output destination (e.g., groups of receivers). Decisions as to where content streams are to be sent and which groups of servers are to receive the streams can be predefined and indicated to the transport manager 170 as a configuration file or XBM function call in real-time. This information can also be entered via a graphical user interface (GUI) 172 or command line utility. In any event, the information is stored in a local database 174. The database 174 also stores information for respective streams relating to defined maximum and minimum IP address and port ranges, bandwidth usage, groups or communities intended to receive the streams, network and stream names, as well as information for user authentication to protect against unauthorized use of streams or other distributed data.

Please amend the paragraph beginning on line 17 of page 17 of the application to read as follows:

Media resource requests from a client ~~22~~ 20 can occur inside of an RTSP connection or via a simple HTTP request. Responses to media resource requests can be metafiles, but can also occur inside of a binary file or via the protocol being used between the client ~~22~~ 20 and server (e.g., server 14, 16 or 18). In each of these instances, responses are similar to a response served by an Internet client-server application to allow sending links to a resource rather than having to send the resource itself. These files and/or response information indicate to the client ~~22~~ 20 the location of requested media, i.e., where it should connect to

and in what order. In video serving applications, metafiles allow content providers to create playlists of video clips, but metafiles can also be used to help define events and other information such as the author or resource owner. Further, the contents of the metafiles can be written as more of a scriptable language to handle conditions and other more dynamic needs. In other words, RTSP can enable encoders 134, receivers 144, and servers or data centers 14, 16 and 18 to communicate with each other to allow for routing, conditional access (e.g., authentication) and bandwidth control in the distribution network 12, in accordance with the present invention. For example, standard RTSP proxies can be provided between any network components to allow them to communicate with each other.

Please amend the paragraph beginning on line 18 of page 18 of the application to read as follows:

The testing component 27 is operable to rewrite the metafile to deny service to clients that request higher bandwidth content than the priority or low-bandwidth connection for which they have paid. Since metafile qualities can be supported in an RTSP channel, the testing component can use a metafile as one type of response, or employ an RTSP comment, if supported, to deny, redirect, and so on, for a response. The testing component 27 is therefore useful to prevent relatively low-speed modems from requesting multicast feeds, for example, and thereby flooding the network. Service providers can also use the testing component 27 to restrict available streams according to client subscription type. A proxy for video requests is one way to make this transparent. Routers and switches in the system 10 can allow an application to reside between a client ~~22~~ 20 and serve in a manner to allow the testing component 27 of the present invention to review client-server communications and make changes as needed.